Perencanaan Rute Gerak Mobile Robot Berpenggerak Differensial Pada Medan Acak Menggunakan Algoritma A* Dikombinasikan Dengan Teknik *Image Blurring*

Ahmad Nashrul Aziz[1], Endra Pitowarno[2]
Jurusan Teknik Elektronika[1], Mekatronika[2]
Politeknik Elektronika Negeri Surabaya
Kampus ITS Keputih Sukolilo Surabaya 60111
Telp. (+62)31-5947280 Fax (+62) 31-5946114
Email: nashrul@nashrul.co.cc[1], epit@eepis-its.edu[2]

Abstrak: Pengembangan teknik otomasi pergerakan robot untuk dapat beroperasi di dunia nyata sudah menjadi bahan penelitian bagi pengembangan mobile robot di dunia saat ini. Untuk dapat mencapai suatu posisi yang diinginkan, mobile robot membutuhkan suatu sistem navigasi yang dapat mengarahkan mobile robot tersebut ke posisi yang diinginkan.

Pada penelitian ini membahas tentang perencanaan rute (*path planning*) pada sebuah model BMP yang mengilustrasikan area kerja robot, *trajectory generation* (pembentukan lintasan). Perencanaan rute dilakukan untuk mendapatkan informasi rute tercepat yang akan dilalui mobile robot, dengan menggunakan algoritma A* yang dikombinasikan dengan teknik *image blurring*. Teknik *image blurring* disini digunakan untuk memperbesar halangan (*obstacle*), sehingga nantinya didapatkan rute yang aman, yaitu rute yang bebas benturan (*collision free*).

Kata kunci: Mobile robot, Algoritma A*, *Image blurring*, *Path planning*, *Trajectory generation*

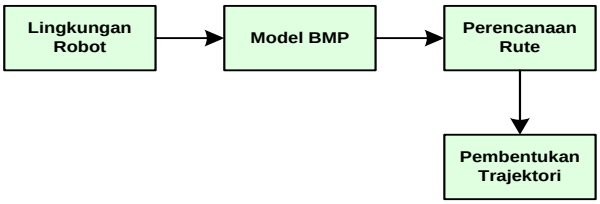
1. Pendahuluan

Salah satu jenis mobile robot yaitu mobile robot berpenggerak differensial atau *Differentially Driven Mobile Robot* (DDMR), adalah suatu mobile robot yang menggunakan dua buah roda penggerak yang independent, sehingga gerakan translasi maupun rotasi robot dihasilkan dari kombinasi pergerakan dua buah aktuator, supaya bisa stabil maka ditambah sebuah roda bebas (*omniderectional*) yang biasa disebut roda *castor*.

Pemakaian peta (model) yang mendeskripsikan ruang kerja robot, pernah dilakukan oleh para peneliti [1], [2], [3]. Perencanaan rute disini diharapkan mampu menghasilkan pergerakan robot yang bebas dari benturan dengan obstacle, selama pergerakan robot dari posisi start sampai goal.

Proses yang dilakukan pada perencanaan rute adalah melakukan proses pengolah model atau peta yang berupa gambar BMP, pertama-tama model yang ada di filter dengan *low pass filter* kemudian dikonversi kedalam format Biner supaya perbedaan antara area kosong dan *obstacle* semakin besar, setelah itu barulah dilakukan proses pencarian rute terpendek (*Shortest path*), dengan menggunakan algoritma A*, setelah itu dilakukan proses pembentukan trajektori refferensi yang nantinya akan dipergunakan sebagai input pergerakan robot.

Program perencanaan rute, dan pembentukan trajektori refferensi dibuat dengan GUI *(Graphic User Interface)* MATLAB. Ilustrasinya dapat dilihat pada gambar 1.



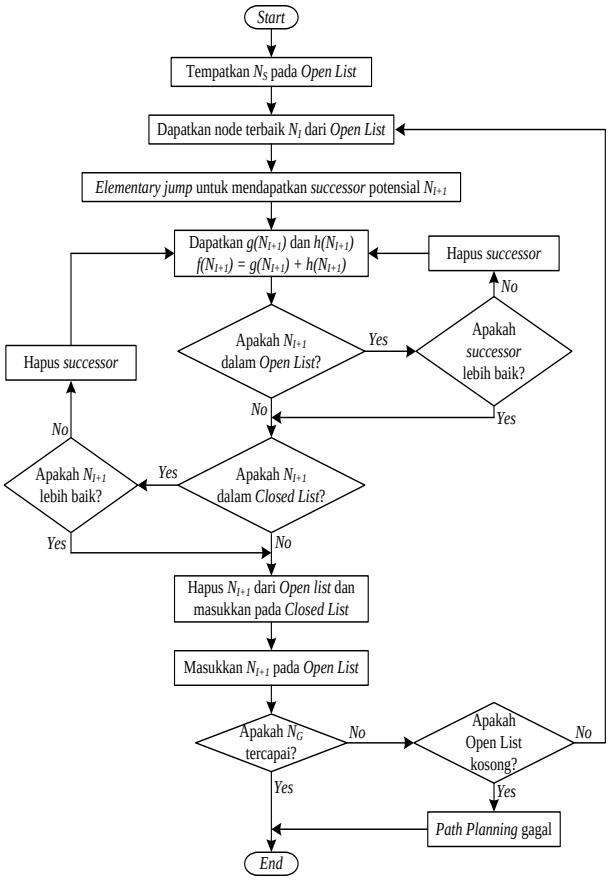
Gambar 1. Blok diagram utama dalam penelitian

1.1 Algoritma A*

A* yaitu merupakan salah satu algoritma yang menggunakan metode heuristik, dimana algoritma ini merupakan gabungan antara algoritma depth-first dan Dijkstra, A* bisa diaplikasikan untuk pencarian jalur yang bebas benturan (*collision-free*). Didalam A* terdapat dua buah himpunan keanggotaan yaitu *Open List* dan *Closed List*, *Open List* adalah kumpulan node yang akan diuji, pada kondisi awal *Open List* adalah node yang mengandung node Start, sedangkan *Closed List* adalah kumpulan node yang telah diuji, pada kondisi awal nilai dari *Closed List* adalah kosong. Setiap node menyimpan pointer node-node sejenis dan terdekat, sehingga keberadaan node mudah dilacak., langkah-langkah dalam A* adalah sebagai berikut [4]:

1. Tempatkan node start N_S pada *Open List*.
2. Jika *Open List* kosong, maka keluar (proses gagal).

3. Keluarkan dari *Open List* dan tempatkan pada *Closed List* sebuah node N_I yang memiliki fungsi cost, $f(N_I)$ minimum.
4. Jika N_I adalah node Goal N_G, maka keluar (proses berhasil) dengan hasil rute diambil dari rekonstruksi (proses balik), pointer dari N_I sampai N_S.
5. Jika sebaliknya, maka perluas N_I, buat semua *successor*, dan letakkan pada pointer balik ke N_{I-1}. untuk setiap *successor* N_{I+1} :
   5.1. Jika N_{I+1} tidak ada dalam *Open List* atau *Closed List*, maka fungsi heuristik, $h(N_{I+1})$ adalah dihitung sebelum total cost proses dari N_I sampai N_{I+1} didapat.
   5.2. Jika N_{I+1} telah berada dalam *Open List* atau *Closed List*, maka pointernya langsung menuju rute yang merupakan hasil terendah $g(N_{I+1})$.
   5.3. Jika N_{I+1} memerlukan penyesuaian pointer dan itu berada dalam *Closed List*, maka node tersebut kembali dibuka.
6. Kembali ke langkah 2.



Gambar 2: *Flowchart* algoritma A*

Flowchart algoritma pencarian heuristik A* bisa diilustrasikan seperti pada gambar 2. Dari langkah 5.1, total cost, $f(N_{I+1})$ dapat dihitung dengan menggunakan persamaan berikut:

$$f(N_{I+1}) = g(N_{I+1}) + h(N_{I+1}) \text{ , dan} \qquad (1)$$

$$g(N_{I+1}) = g(N_I) + c(N_I, N_{I+1}) \qquad (2)$$

Dalam persamaan (2), $c(N_I, N_{I+1})$ adalah cost proses perpindahan dari N_I menuju N_{I+1}. Dalam hal ini fungsi heuristik $h(N_{I+1})$ didapat dari jarak *Euclidean* diantara pengganti, N_{I+1} dan node Goal, N_G. perhitungan fungsi $h(N_{I+1})$ disajikan dalam persamaan berikut:

$$h(N_{I+1}) = \sqrt{(x_{N_{I+1}} - x_{NG})^2 + (y_{N_{I+1}} - y_{NG})^2} \qquad (3)$$

1.2 *Low pass filter* (*Image blurring*)

Low pass filter merupakan salah satu metode filter yang yang dapat digunakan untuk mengurangi noise yang ada pada gambar, karena keberadaan noise dapat menjadi gangguan untuk proses selanjutnya, dan untuk filter jenis low pass filter ini pada prinsipnya adalah filter yang mengambil nilai tengah dari sampel matrik yang diberikan, sebagai contoh diambil matrik 3x3 dan dari matrik tersebut terdapat 9 nilai kombinasi warna RGB yang berbeda-beda dan keberadaannya tidak berurut sehingga dilakukan proses mengurutkan 9 nilai tersebut dari yang terkecil ke yang terbesar, lalu diambil nilai tengah dari urutan tersebut.

Dalam image processing, *low pass filter* biasa disebut blurring. Ada banyak jenis *low pass filter* yang dapat digunakan, dilihat dari bentuk dan derajat filternya. Filter-filter tersebut menggunakan suatu kernel tertentu dalam bentuk window matrik 2D dengan ukuran tertentu. Operasi low pass filter dilakukan dengan mengganti intensitas suatu pixel dengan rata-rata nilai pixel tersebut dengan nilai pixel-pixel tetangganya, bentuk dasar dari low pass filter adalah sebagai berikut :

$$H=\begin{bmatrix} H_{0,0} & H_{0,1} & \cdots & H_{0,K-1} \\ H_{1,0} & H_{1,1} & \cdots & H_{1,K-1} \\ \cdots & \cdots & \cdots & \cdots \\ H_{B-1,0} & H_{B-1,1} & & H_{B-1,K-1} \end{bmatrix} \quad Dengan \sum_{b=0}^{B-1}\sum_{k=0}^{K-1} H_{b,k} = 1 \qquad (4)$$

1.3 Konversi RGB ke Biner

Representasi data dalam bentuk biner banyak digunakan jika yang diperlukan adalah informasi mengenai ada atau tidaknya suatu obyek pengamatan tertentu. Ada atau tidaknya dapat berasal dari perbedaan level intensitas (lebih terang atau lebih gelap) dan dapat berasal dari nilai ambang batas (gelap atau terang). Format biner disimpan dalam bentuk data 8 bit (1 Byte).

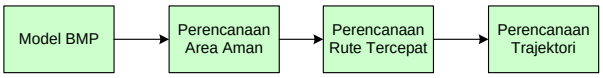
$$H_{B,K} = \{0,1\} \; atau \; \{0,255\} \qquad (5)$$

Untuk melakukan pengubahan dari format RGB ke biner dapat menggunakan metode thresholding, artinya jika nilai RGB lebih besar dari nilai tertentu akan dianggap 1 (atau 255) dan sebaliknya akan dianggap 0.

$$Biner = \begin{cases} 0 & jika\ RGB < Threshold \\ 1\ atau 255 & jika\ RGB \geq Threshold \end{cases} \qquad (6)$$

2. Perencanaan Rute Pergerakan Robot

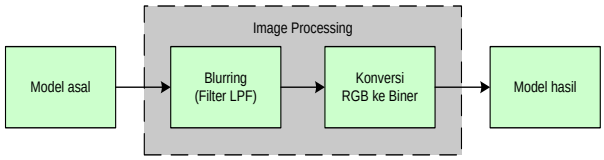
Perencanaan rute gerak robot menghasilkan rute dari titik start ke target, yaitu berupa rute aman dan tercepat tanpa bersinggungan dengan halangan (*collision free*) pada sebuah model BMP, model BMP disini adalah sebuah file gambar dengan ukuran 160x160 pixel yang mengilustrasikan area kerja robot dengan ukuran 160x160 cm. Langkah-langkah yang dilalui diperlihatkan pada gambar 3.



Gambar 3: Diagram blok perencanaan rute

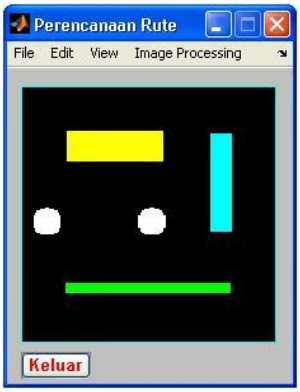
2.1 Perencanaan Area Aman

Untuk mendapatkan area aman pada model dilakukan image blurring dengan menggunakan filter LPF, serta image biner. Proses pada perencanaan area aman dapat dilihat pada blok diagram pada gambar 4.



Gambar 4: Diagram blok perencanaan area aman

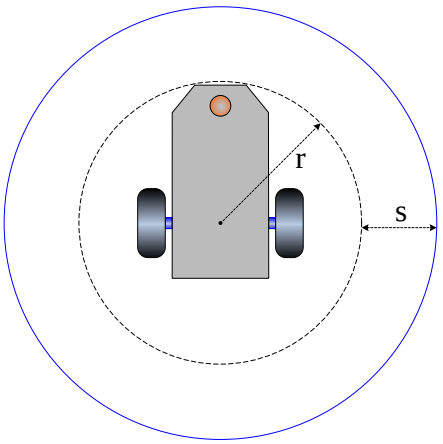
Pada model asal, pixel dengan warna hitam yaitu pixel dengan komposisi warna (R=0, G=0, B=0) diartikan sebagai area kosong, sedangkan pixel dengan warna selain hitam diartikan sebagai halangan (*obstacle*). Model asal diperlihatkan pada gambar berikut.



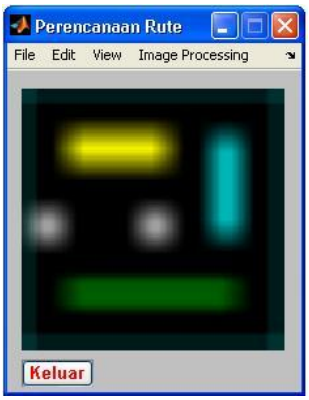
Gambar 5: Model asal

Dimensi robot yang dipergunakan dalam proses perencanaan rute diperlihatkan pada gambar 6, dimana r adalah jarak antara titik tengah as roda dengan bagian terluar dari fisik robot yaitu sebesar 11 cm, s adalah *safety margin*, dalam hal ini ditentukan sebesar 9 cm.

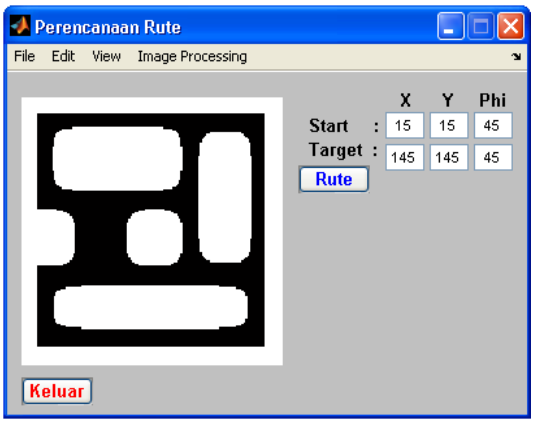
Proses blurring disini berfungsi untuk memperbesar ukuran obstacle supaya robot tidak berbenturan dengan *obstacle*, yaitu dengan nilai threshold sebesar 20, didapat dari penjumlahan antara jarak r dengan *safety margin* (s), selanjutnya nilai tersebut dipakai untuk menentukan ukuran matrik 2D yang akan digunakan pada proses blurring, sehingga ukuran matrik yang dipakai yaitu 20x20. Setelah dilakukan proses blurring, maka model akan berubah seperti diperlihatkan pada gambar 7, pada gambar tersebut obstacle menjadi lebih kabur, karena terjadi efek pemerataan derajat keabuan, sehingga gambar yang diperoleh tampak lebih kabur kontrasnya, terutama pada bagian tepi. Selanjutnya supaya obstacle dan area kosong lebih tampak perbedaannya maka, dilakukan proses konversi warna dari RGB ke Biner.



Gambar 6: Dimensi robot



Gambar 7: Model setelah dilakukan proses blurring

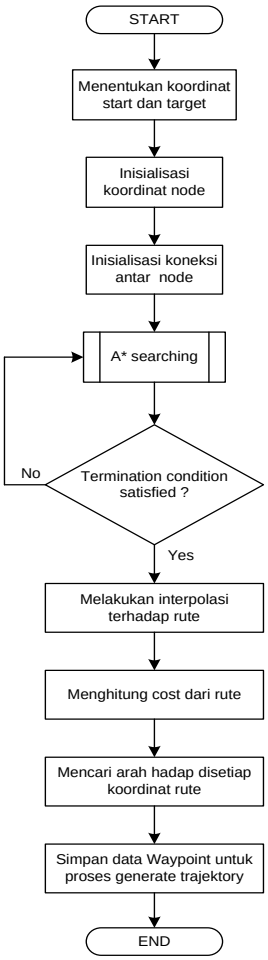


Gambar 8: Model setelah dilakukan proses konversi RGB ke biner

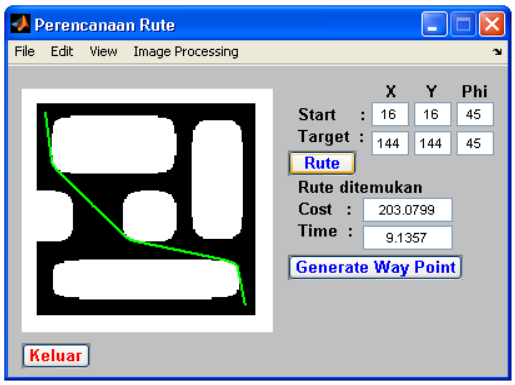
Setelah dilakukan proses konversi RGB ke Biner, maka model akan berubah seperti diperlihatkan pada gambar 8. Untuk melakukan pengubahan dari format RGB ke Biner yaitu dengan menggunakan metode thresholding, nilai yang dipakai pada proses ini adalah 0,025, artinya jika nilai RGB pada suatu pixel lebih besar dari 0,025 akan dianggap 1 dan sebaliknya akan dianggap 0. Kumpulan pixel yang bernilai 0 atau berwarna hitam dianggap sebagai area kosong, sedangkan kumpulan pixel yang bernilai 1 atau berwarna putih dianggap sebagai obstacle.

2.2 Perencanaan Rute Tercepat

Pada bagian ini akan dibahas tentang proses pencarian rute tercepat dengan menggunakan algoritma A*, dan melakukan proses interpolasi pada titik-titik koordinat hasil dari rute yang didapat, supaya rute yang terbentuk akan menjadi lebih halus, kemudian menghitung cost dari rute yang telah ditemukan, serta mencari arah hadap robot disetiap koordinat rute yang telah ditemukan. Proses pada perencanaan rute bisa dilihat pada gambar 9. Setelah dilakukan pemrosesan filter low pass dan konversi RGB ke biner, selajutnya akan dilakukan proses pencarian rute tercepat, dengan terlebih dahulu menentukan koordinat titik start dan target pada area model yang berwarna hitam (area kosong).



Gambar 9: *Flowchart* perencanaan rute



Gambar 10: Hasil pencarian rute

Setelah didapatkan data rute yaitu data koordinat rute, selanjutnya dilakukan interpolasi dengan menggunakan metode *Nearest neigbor*, ini dilakukan supaya data koordinat rute semakin bervareasi, atau bisa dikatakan rute yang dihasilkan akan menjadi lebih halus. Pemakaian metode *Nearest neigbor* dilakukan karena metode yang lain, misalnya *Cubic spline* mengakibatkan komputasi menjadi lebih berat, sehingga membutuhkan waktu yang lama, serta *resources* komputasi yang lebih baik. Hasil data pencarian rute diperlihatkan pada gambar 10. Nilai total cost dari start ke target dapat dihitung dengan cara, terlebih dahulu mencari ukuran dari matrik data rute x atau y, setelah didapatkan informasi ukuran, kemudian mencari jarak setiap koordinat node, dari start sampai ke target, total cost adalah jumlah cost dari semua node.

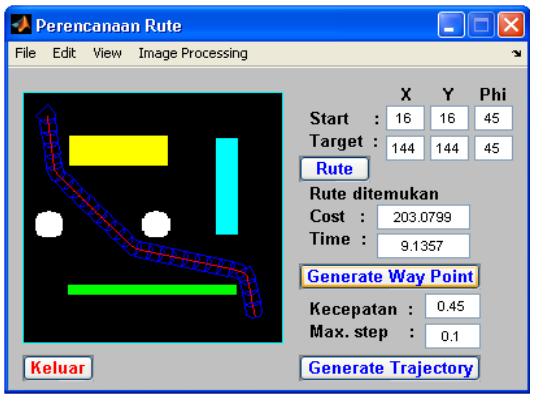
2.3 Perencanaan Trajektory

Perencanaan trajektory dilakukan untuk mendapatkan rute gerak bagi robot, rute yang dimaksud disini yaitu berupa data posisi dan kecepatan yang mengandung informasi waktu, atau dengan kata lain berupa perintah gerak pada setiap step pergerakan robot, yang selalu berubah setiap saat, sesuai dengan perintah yang diberikan pada robot. Dalam generate trajektory pertama kali yang perlu dilakukan adalah, melakukan pengaturan *solver*, yaitu suatu fungsi untuk membangkitkan clock yang akan dipakai pada proses pembentukan trajektory, serta dipakai untuk timing pergerakan robot. Yang perlu diperhatikan dalam pengaturan *solver* yaitu pemilihan tipe step, pada proses perencanaan trajektory yaitu menggunakan tipe variabel step, dengan memakai *solver ode45(Dorman-Prince)*, pada tipe ini perlu dilakukan pengaturan nilai step, antara lain step maksimal, step minimal, step awal, dan tingkat akurasi solver.

3. Data Perencanaan Rute dan Analisa Hasil

Untuk mendapatkan arah hadap robot dilakukan dengan cara mencari nilai sudut dari jarak antara koordinat pada setiap node, yaitu dengan cara mencari nilai akar tangen, dengan terlebih dulu mencari jarak antar koordinat baik koordinat x maupun y, rute yang diperoleh setelah proses generate waypoint

diperlihatkan pada gambar 11, dalam gambar tersebut terlihat step-step pergerakan robot, yaitu model robot yang berwarna biru, garis merah adalah titik-titik lintasan referensi robot.



Gambar 11: Rute yang dihasilkan setelah melalui proses generate waypoint

Tabel 1: Data *waypoint* hasil dari proses perencanaan rute

Node	X	Y	dX	dY	Phi (rad)
1	16	16	-	-	0.7854
2	16.0603	16.4908	0.0603	0.4908	1.4486
3	16.1206	16.9817	0.0603	0.4908	1.4486
. . .	. . .	. . .	. . .	. . .	. . .
24	17.3868	27.2894	0.0603	0.4908	1.4485
25	17.4471	27.7802	0.0603	0.4908	1.4485
26	17.5075	28.2710	0.0603	0.4908	1.4485
. . .	. . .	. . .	. . .	. . .	. . .
250	87.1322	104.5782	0.4822	0.1030	0.2104
251	87.6144	104.6812	0.4822	0.1030	0.2105
252	88.0966	104.7844	0.4822	0.1030	0.2106
. . .	. . .	. . .	. . .	. . .	. . .
355	137.1484	116.4901	0.4121	0.2261	0.5019
356	137.5006	116.7986	0.3522	0.2996	0.7049
357	137.8349	117.1165	0.3343	0.3179	0.7603
. . .	. . .	. . .	. . .	. . .	. . .
412	143.8200	143.0347	0.0899	0.4826	1.3865
413	143.9100	143.5174	0.0899	0.4826	1.3865
414	144	144	0.0899	0.4826	0.7854

Proses pembentukan model robot dilakukan dengan cara memakai fungsi element pembentuk model tersebut, yaitu fungsi garis lurus baik horisontal maupun maupun vertikal, dan fungsi garis miring, disemua kuadran. Proses ini dilakukan tiap 15 node, koordinat dan arah hadap dari model robot selalu berubah sesuai dengan data posisi pada tiap step pergerakan. Dari percobaan perencanaan rute dengan posisi start (16,16,45) dan posisi target (144,144,45) cost yang didapat sebesar: 203,0799 dan waktu komputasi selama: 9,1357 detik, serta jumlah *node* yang didapatkan sebanyak 414 *node*, data *waypoint* yang didapat diperlihatkan pada tabel 1.

4. Kesimpulan

Dari seluruh proses yang telah dibahas pada penelitian ini bisa diperoleh kesimpulan sebagai berikut:

1. Dengan melakukan image blurring terhadap model dan melakukan konversi ke bentuk biner, terbukti mampu menghasilkan rute yang aman, tanpa bergesekan dengan *obstacle* yang ada pada model (*collision free*).
2. Algoritma A* terbukti mampu menghasilkan rute yang pendek sehingga robot membutuhkan waktu yang singkat dalam melakukan penelusuran rute (*tracking*).
3. Proses interpolasi *nearest neigbor* memiliki pengaruh terhadap rute, yaitu rute yang dihasilkan menjadi lebih halus dan akurat, selain itu waktu komputasi menjadi lebih singkat, serta tidak perlu membutuhkan *resource* komputasi yang lebih bagus.

5. Referensi

[1] De Luca, A., Oriolo, G., and Samson, C. (1998). "*Feedback control of a nonholonomic car-like robot*", in *Robot Motion Planning and Control*, J.-P. Laumond (Ed.). Lecture Notes in Control and Information Sciences, 229, pp. 171–253, Springer Verlag, London.

[2] Atyia, S., Hager, G. D. (1993). *Real-time vision-based robot localization*. IEEE Trans. on Robotics and Automation, 9(6), pp. 785-800.

[3] Durrant-Whyte, H. F. (1994). *Where am I? A tutorial on mobile vehicle localization*. Industrial Robot, 21(2), pp. 11-16, (1994).

[4] Mailah M., Jamaluddin H., Pitowarno E., Purnomo D. S., Hing T. H., and Rahim M. A. B. A. (2007). *Intelligent Material Handling Mobile Robot for Industrial Purpose with Active Force Control Capability*. Universiti Teknologi Malaysia: Laporan Akhir Penyelidikan.